An Energy-Constrained Invader Depth-First Search Algorithm for N-ary Tree Traversal

Bui Phi Hung[1] and Nguyen Minh Tuan[1*]

[1*]Faculty of Information Technology, Posts and Telecommunications Institute of Technology, 11 Nguyen Dinh Chieu, Sai Gon ward, 700000, Ho Chi Minh city, Viet Nam.

*Corresponding author(s). E-mail(s): minhtuan@ptit.edu.vn;
Contributing authors: n23dcat028@student.ptithcm.edu.vn;

Abstract

Depth First Search (DFS) is a fundamental algorithm in mathematical discrete and data structures. Different variants of DFS exploration have been created and deployed in applied science, such as Piecemeal-DFS. This study proposes an efficient algorithm, namely Invader DFS (IDFS). IDFS is an energy-constrained exploration algorithm in which a DFS traversal $\boldsymbol{R}_{\text{DFS}} = (\boldsymbol{v}_{0j}, \ldots, \boldsymbol{v}_{lj})$ is divided into routes $\boldsymbol{R}_{1j}, \ldots, \boldsymbol{R}_{kj}$, each starting at the root, continuing from the last vertex of $\boldsymbol{R}_{i-1,j}$, and returning to the root to recharge the full energy when this limit is reached. Specifically, in this study, the DFS algorithm is implemented to investigate a forest of $\boldsymbol{n}$ weighted trees and unify the routes in the first tree. The process is refined by using the residual after each exploration route to discover additional branches and edges, thereby increasing the number of routes and reducing the cost of exploration. The analysis shows that this algorithm optimizes the number of routes and the total cost compared to exploring each tree individually. When applied to a forest of $\boldsymbol{n}$ trees, this function, usually illustrated by robots, expends a certain amount of exploration energy to establish routes and the exploration cost.

Keywords: Piecemeal DFS, DFS, Tree Exploration, Energy Constraint

1 Introduction

Graph exploration plays an important role in graph theory, underpinning applications in network analysis and robotics. The DFS and Breadth-First Search (BFS) algorithms form the basis for numerous different variants to specific structural or resource-constrained needs. Piecemeal DFS (PDFS) partitions a depth-first traversal into an energy-constrained, overweighted tree. The method is proven correct under a given energy bound B, with additional theoretical guarantees on the number of routes and total cost of the route, as detailed in [1]. Tuan et al. [2] extend this framework by proposing a PDFS-based exploration strategy for two predefined weighted trees, which are merged into a unified exploration task under global energy constraints. Iterative Deepening DFS (IDDFS) [3] combines the memory efficiency of DFS, with a space complexity of O(d), and the completeness of BFS in unweighted graphs, by repeatedly applying depth-bounded searches.

Similarly, depth-limited DFS [4] introduces a hard cutoff at depth L to prevent infinite descent in acyclic or infinite graphs; Bampas et al. [5] address the problem of exploring an unknown weighted tree using multiple mobile agents with bounded energy. Their goal is to maximize the number of visited vertices, and they present an online algorithm with a competitive ratio of 3 compared to the optimal offline solution. In a related direction, Dutta and Sharma [6] proposed an energy-constrained DFS algorithm for online coverage path planning in unknown environments. Their method provides constant-factor approximation guarantees relative to the shortest possible path, enabling efficient exploration under tight energy budgets.

Further advances include reducing the auxiliary space from $\mathcal{O}(n \log n)$ bits to $\mathcal{O}(n \log \log n)$ bits while maintaining $\mathcal{O}(n+m)$ time complexity [7], and interleaving forward and backward DFS in parallel to compute strongly connected components in linear time [8]. Currently, no work addresses exploring a forest of n trees and optimizing the transfer of residual energy between paths to minimize the number of paths. In this paper, IDFS is deployed to explore forests of n trees through a virtual root structure and use the surplus to explore additional branches or edges, thereby reducing both the number of visits and the total exploration cost.

2 Methods

This section will apply IDFS for exploring n trees by optimizing the energy to obtain possible trees and combining them into the first tree. The following basic concepts are related to this project as follows:

2.1 Fundamental concepts

Definition 1 A forest is a set of trees with no cycle. In our case, the exploration forest consists of n discrete trees once they have been merged or linked. Route cost for a route $R_{ij} = (v_{01},\ \ldots, v_{n1}, v_{02}, \ldots, v_{n2}, \ldots, v_{0n},\ v_{1n},\ \ldots, v_{nn})$, and the sum of edge-weights along the route is shown as follows:

$$w(R) = \sum_{i=0}^{n} \sum_{j=1}^{n} w_{ij},$$

where $w(R)$ is the total cost of the route, and w_{ij} is the weight of edge i^{th} corresponding to j^{th}-tree.

An exploration strategy S consists of a sequence of routes R_{ij} with $i = 1, \ldots, k, j = 1, \ldots, n$, i.e., $S = (R_{1j}, R_{2j}, \ldots, R_{kj})$, where each $R_{ij} \leq B$. The number of routes is denoted as $|S| = k$.

Exploration cost is defined as:

$$\xi(S) = \sum_{i=1}^{k} \sum_{j=1}^{n} \xi(R_{ij}).$$

In other words, this is the total cost of the IDFS strategy.

Definition 2 Let the forest $F = T_1 \cup T_2 \cup \ldots T_n$, F is a set of n trees T_i. Therefore, considering the forest F as a large tree, denoted T_F. Some related properties are presented as follows:

- The total cost of the IDFS strategy on the forest F, denoted as $\xi(IDFS(T_F))$.
- The optimal strategy is namely $COPT(T_F)$.
- Edge weight $\omega(e)$ is the length of an edge e; the total weight of a subtree T'_F is $\omega(T'_F)$.
- The potential of a vertex v is $\varphi(v) = B/2 - d(r, v)$, where r is the root of T_F and $d(r, v)$ is the distance from r to v, and for a subtree T'_F, $\varphi(T'_F) = \varphi(v)$, where u, v are virtual roots.
- The adversarial IDFS is denoted AIDFS(T_F), maximizing the cost over the lengths of the first route $B' \leq B$. This is the worst-case IDFS strategy. By this definition, $\xi(\text{AIDFS}(T_F)) \geq \xi(\text{IDFS}(T_F))$ for any tree T_F. Thus, $\xi(\text{AIDFS}(T_F))$ implies the bound for $\xi(\text{IDFS}(T_F))$.
- Heavy/Light subtrees: A subtree T'_F is light if $\omega(T'_F) \leq \varphi(T'_F)$, and a subtree is called T'_F heavy if $\omega(T'_F) > \varphi(T'_F)$ shown in Fig. 1.
- Heavy degree: A vertex v is called heavy if the subtree $T_F[v]$ rooted at v is heavy, and heavydeg(v) is denoted as the number of heavy downward edges from vertex v. This function measures the number of heavy subtrees attached to v, illustrated in Fig. 2.
- For a skinny tree (ST), a vertex v heavydeg$(v) = 1$, the path from v to a descendant r' where each internal vertex has a heavy degree 1 has at most one light edge incident to each vertex. $\vartheta(T_F)$ is the minimum weight of edges incident to the leaves; otherwise, $\vartheta(T_F) = 0$, it appears in the final cost bounds.

Theorem 1 (Bound on explored cost) *Given a forest T_F where the maximum distance from the virtual root to any leaf is at most $B/2$, we have:*

$$\xi(IDFS(T_F)) \leq 12 \cdot \xi(COPT(T_F)),$$

where $\xi(COPT(T_F))$ denotes the minimum total cost of any exploration strategy.

Proof The subtrees are classified as heavy or light, and some auxiliary lemmas are proved that bound the exploration cost in each case. Using a key inductive result in Theorem 2 applied to any heavy tree, an $IDFS$ exploration never incurs more than 12 times the cost of an optimal strategy. After that, a tree rearrangement technique is described in which, given an arbitrary tree T_F, it constructs a modified tree T_ε that satisfies a special structural property.

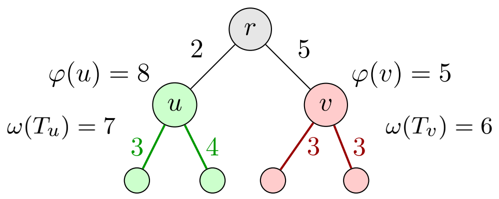
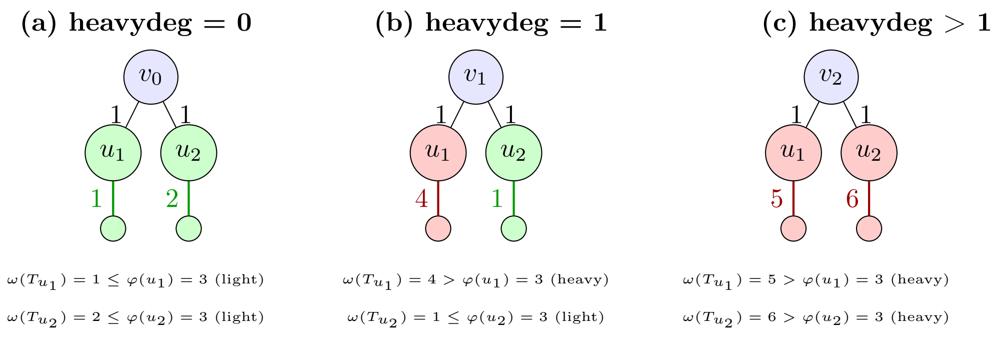


Fig. 1 Light (green) and heavy (red) subtrees with weights and potentials (B=20).



(a) heavydeg = 0
(b) heavydeg = 1
(c) heavydeg > 1
v_0
v_1
v_2
u_1
u_2
$\omega(T_{u_1}) = 1 \le \varphi(u_1) = 3$ (light)
$\omega(T_{u_2}) = 2 \le \varphi(u_2) = 3$ (light)
$\omega(T_{u_1}) = 4 > \varphi(u_1) = 3$ (heavy)
$\omega(T_{u_2}) = 1 \le \varphi(u_2) = 3$ (light)
$\omega(T_{u_1}) = 5 > \varphi(u_1) = 3$ (heavy)
$\omega(T_{u_2}) = 6 > \varphi(u_2) = 3$ (heavy)

Fig. 2 The classification of three cases with `heavydeg(v)`: 0, 1, and >1, with $B = 10$, $d(r, v_i) = 1$. $\varphi(u_i) = \frac{B}{2} - d(r, u_i) = \frac{B}{2} - (d(r, v_i) + d(v_i, u_i))$. Ex: (a) $\varphi(u_1) = 5 - (1 + 1) = 3$.

It is shown that if the inductive cost bound holds for T_ε, it also holds for T_F (Lemma 4), thereby allowing us to assume the special structure without loss of generality. □

For any vertex v, let $e_1, e_2, \dots, e_k$ be all the downward edges from v_{ij} (the edges from v to each of its children at the specific tree j). Let $T[e_{ij}]$ denote the subtree consisting of edge e_{ij} together with the entire subtree rooted at the lower endpoint of e_{ij}. Then some lemmas are derived as follows:

Lemma 1 (Cost decomposition across children) *For any node $v = v_{ij}$ with downward edges $e_{1j}, \dots, e_{kj}$,*

$$\begin{aligned} (i) \quad & \xi(COPT(T[v])) = \sum_{i=1}^{k}\sum_{j=1}^{n} \xi(COPT(T[e_{ij}])), \\ (ii) \quad & \xi(AIDFS(T[v])) \le \sum_{i=1}^{k}\sum_{j=1}^{n} \xi(AIDFS(T[e_{ij}])). \end{aligned}$$

Proof For part (i), an optimal exploration of $T[v]$. Each route of $COPT(T[v])$ explores a subset of the children subtrees $T[e_{ij}]$ and then returns to v. An optimal strategy can be structured to explore each child subtree $T[e_{ij}]$ independently: for each i, take the portion of each route that is spent in subtree $T[e_{ij}]$, and append those portions together (adding a return to v at the end) to form a valid B-bounded route covering $T[e_{ij}]$. Doing this for each child e_{ij} yields a set of disjoint routes covering each $T[e_{ij}]$ with no increase in total number of routes or distance traveled. Therefore, $\xi(COPT(T[v]))$ splits as the sum of optimal costs for each subtree $T[e_{ij}]$. This establishes (i).

For part (ii), any AIDFS exploration of $T[v]$. We can concatenate all its routes (in order) into one sequence. Now, cut this sequence whenever the traversal returns to v. By doing so, we partition the $AIDFS$ routes into groups such that each group corresponds to exploration within a single child subtree $T[e_{ij}]$. Thus, we obtain, for each child edge e_{ij}, a subsequence of the adversarial routes that collectively explore $T[e_{ij}]$ and never visit v except at the start and end. This subsequence of routes is itself a feasible DFS-based exploration of the subtree $T[e_{ij}]$. Therefore, $\xi(AIDFS(T[v]))$ is equal to the sum of the costs of some feasible DFS explorations of each $T[e_{ij}]$. Taking the maximum cost DFS strategy for each $T[e_{ij}]$ can only increase this sum, which proves (ii). □

Lemma 2 *If T_F is light, then $\xi(AIDFS(T_F)) \le 2\,\xi(COPT(T_F)) - \vartheta(T_F)$.*

Proof If T_F is light ($\omega(T_F) \le \varphi(T_F)$), AIDFS($T_F$) either:

+ Consists of one route, with $\xi(\text{AIDFS}(T_F)) = \xi(\text{COPT}(T_F))$.

+ Consists of two routes, where the first omits one edge to a leaf, with length at most $2\omega(T_F) - \vartheta(T_F)$, and the second covers the remaining vertices, with length at most $2\varphi(T_F)$.

Thus:

$$\xi(\text{AIDFS}(T_F)) \le 2\omega(T_F) + 2\varphi(T_F) - \vartheta(T_F) \le 4\omega(T_F) - \vartheta(T_F).$$

Since $\xi(\text{COPT}(T_F)) \ge 2\omega(T_F)$ (each edge is traversed at least twice), we have:

$$\xi(\text{AIDFS}(T_F)) \le 2 \cdot \xi(\text{COPT}(T_F)) - \vartheta(T_F).$$

For example, considering the path {r , u , w} with both edges of weight 1. Then $\omega(T_F) = 2$, $\varphi(T_F) = 2$, so T_F is light. Both COPT and AIDFS use one route of cost 4, and indeed $4 \le 2 \cdot 4 - 1$. □

Theorem 2 (Inductive cost bound for heavy trees) *If T_F is a heavy tree with root r, then:*

(i) If $heavydeg(r) = 1$ (i.e., T_F is heavy and satisfies (ST), a skinny heavy tree), then

$$\xi(AIDFS(T_F)) \;\le\; 12\,\xi(COPT(T_F)) \;-\; 10\,\varphi(T_F) \;-\; \vartheta(T_F).$$

(ii) If $heavydeg(r) \ne 1$ (the root has more than one heavy child), then

$$\xi(AIDFS(T_F)) \;\le\; 12\,\xi(COPT(T_F)) \;-\; 20\,\varphi(T_F) \;-\; \vartheta(T_F).$$

Proof We prove this by induction on the number of heavy edges in T_F.

Case 0: *$heavydeg(r) = 0$.* In this case T_F is light, and Lemma 2 implies

$$\xi(AIDFS(T_F)) \le 2\,\xi(COPT(T_F)) - \vartheta(T_F).$$

Since T_F is light we have $\varphi(T_F) \ge \omega(T_F)$, and also $\xi(COPT(T_F)) \ge 2\,\omega(T_F)$. Therefore

$$\begin{aligned}\xi(AIDFS(T_F)) &\le 2\xi(COPT(T_F)) - \vartheta(T_F)\\ &\le 2\xi(COPT(T_F)) - \vartheta(T_F) \le (12\xi(COPT(T_F)) - 20\omega(T_F)) - \vartheta(T_F).\end{aligned}$$

Corresponding to $\xi(COPT(T_F)) \geq 2\omega(T_F)$, we have $12\xi(COPT(T_F)) - 20\omega(T) \geq 0$. Simplifying, we get

$$\xi(AIDFS(T_F)) \leq 12\xi(COPT(T_F)) - 20\omega(T_F) - \vartheta(T_F).$$

Similarly, $\xi(COPT(T_F)) \geq 2\omega(T_F)$, we get $2\xi(COPT(T_F)) \geq 4\omega(T_F)$, and $4\omega(T_F) \leq 12\xi(COPT(T_F)) - 20\omega(T_F)$. Moreover, $2\xi(COPT(T_F)) \leq 12\xi(COPT(T_F)) - 20\omega(T_F)$. Thus, we get:

$$\xi(AIDFS(T_F)) \leq 12\xi(COPT(T_F)) - 20\omega(T_F) - \vartheta(T_F).$$

This establishes the desired inequality in the base case. For the inductive step, assume the claim holds for all heavy proper subtrees of T_F. We consider two cases, according to the heavy degree of the root r.

Case 1: *$heavydeg(r) > 1$.* For each light subtree $T[e'_{ij}]$, by Lemma 2:

$$\xi(\text{AIDFS}(T[e'_{ij}])) \leq 2 \cdot \xi(\text{COPT}(T[e'_{ij}])) - \vartheta(T[e'_{ij}]). \tag{1}$$

Lemma 3 *For each of trees, $j = 1, \ldots, n$, the root has $h \geq 2$ heavy children and some number $l \geq 0$ of light children. Let $e_{1j}, \ldots, e_{hj}$ be the set of heavy child-edges of r, and $e'_{1j}, \ldots, e'_{lj}$ be the set of light child-edges of r. Lemma is derived as follows:*

(i) $\xi(\text{COPT}(T_F)) = \sum_{i=1}^{h} \xi(\text{COPT}(T[e_{ij}])) + \sum_{i=1}^{l} \xi(\text{COPT}(T[e'_{ij}]))$, $j = 1, \ldots, n$.

(ii) $\xi(\text{AIDFS}(T_F)) \leq \sum_{i=1}^{h} \xi(\text{AIDFS}(T[e_{ij}])) + \sum_{i=1}^{l} \xi(\text{AIDFS}(T[e'_{ij}]))$, $j = 1, \ldots, n$.

Combining Theorem 2(i) with Eq 1 and Lemma 3:

$$\begin{aligned}
\xi(AIDFS(T_F)) \;\leq\; & \sum_{i=1}^{h} \Big(12\,\xi(COPT(T[e_{ij}])) - 10\,\varphi(T[e_{ij}]) - \vartheta(T[e_{ij}])\Big) \\
& + \sum_{j=1}^{l} \Big(2\,\xi(COPT(T[e'_{ij}])) - \vartheta(T[e'_{ij}])\Big) \\
= {} & 12\sum_{i=1}^{h} \xi(COPT(T[e_{ij}])) \;+\; 2\sum_{j=1}^{l} \xi(COPT(T[e'_{ij}])) \;-\; 10\sum_{i=1}^{h} \varphi(T[e_{ij}]) \\
& - \sum_{i=1}^{h} \vartheta(T[e_{ij}]) \;-\; \sum_{j=1}^{l} \vartheta(T[e'_{ij}]) \;.
\end{aligned}$$

Since $\varphi(T_F) = \varphi(T[e_{ij}])$(because the subtree $T[e_{ij}]$ has the same root as T_F, or more precisely the same distance from the root) and $h \geq 2$:

$$\sum_{i=1}^{h} (-10 \cdot \varphi(T[e_{ij}])) = -10 \cdot \varphi(T_F) \cdot h \leq -20 \cdot \varphi(T_F).$$

Thus:

$$\xi(\text{AIDFS}(T_F)) \leq 12 \cdot \sum_{i=1}^{h} \xi(\text{COPT}(T[e_{ij}])) + 2 \cdot \sum_{i=1}^{l} \xi(\text{COPT}(T[e'_{ij}])) - 10 \cdot \varphi(T_F) \cdot h - \vartheta(T_F).$$

Using Lemma 3(i), we obtain:

$$\xi(\text{AIDFS}(T_F)) \leq 12 \cdot \xi(\text{COPT}(T_F)) - 20 \cdot \varphi(T_F) - \vartheta(T_F).$$

Satisfies Theorem 2 (ii).

Lemma 4 *For any tree T_F, alway exists $\varepsilon > 0$ and a tree T_ε satisfying (ST) such that if Theorem 2 holds for T_ε, it holds for T_F.*

Case 2: *heavydeg*$(r) = 1$. Let r' denote the closest descendant of r in tree T_F that is classified as heavy and for which heavydeg$(r') \neq 1$. Note that such a vertex always exists, and r' is unique. Denote by P the connecting path from r to r'. Furthermore, we denote $e_1, e_2, \dots, e_l$ as all light edges connected to vertices in $V(P)\backslash\{r\}$, arranged in non-decreasing order according to potential. (Note that the subtree rooted at r' has already been handled in the induction case or in the case where the heavy degree is greater than 1.)

Let $\varphi_i = \varphi(T[e_{ij}])$ for $i \in \{1, \dots, l\}$.

By Lemma 4 (specifically because Lemma 4 ensures that in tree T_F, all edges $e_1, \dots, e_l$ have distinct potentials), we have:

$$\varphi(r) \geq \varphi_l > \varphi_{l-1} > \cdots > \varphi_1 > \varphi(r') = \varphi_0.$$

The path P, which may contain degree-2 vertices, though these are insignificant for analysis and thus omitted, and the positions of edges e_i along with their corresponding potentials. We summarize:

$$w_i = \omega(T[e_{ij}]) = \frac{1}{2}\xi(\mathrm{COPT}(T[e_{ij}])) \quad \text{for } i \in \{1, \dots, l\}; w_0 = \frac{1}{2}\xi(\mathrm{COPT}(T[r'])).$$

Let c be the number of routes in COPT(T_F), and let x_i denote the lowest potential reached by the i^{th} route on path P (ignoring potentials in subtrees). Without loss of generality, assume:

$$x_1 \leq x_2 \leq \cdots \leq x_c.$$

For each $j \in \{1, \dots, c\}$, the first j routes of COPT(T_F) must cover all subtrees $T[e_{ij}]$ where the potential of e_i's higher endpoint is less than x_j; otherwise, some vertices would not be visited by COPT(T_F). The total weight of these subtrees is:

$$\sum_{i:\varphi_i < x_j} w_i.$$

Observe that the total length of all segments of the i^{th} route not lying on path P is at most $2x_i$. Therefore, the total weight of these subtrees satisfies:

$$\sum_{i:\varphi_i < x_j} w_i \leq x_1 + \cdots + x_{j-1} \quad \forall j \in \{1, \dots, c+1\}, \tag{2}$$

with the convention that $x_{c+1} = +\infty$.

□

Now, we aim to bound the cost of AIDFS(T_F) on path P in relation to: $\sum_{i=1}^{c} 2(\varphi(r) - x_i)$, which represents the cost of COPT(T_F) on path P. To accomplish this, we begin by comparing the elements $x_1, \dots, x_c$ with a sequence $y_1, \dots, y_d$ which is selected according to an appropriate greedy strategy.:

$$y_j = \min\left\{ y : \sum_{i:\varphi_i \leq y} w_i > y_1 + \cdots + y_{j-1} \right\},$$
$$d = \min\left\{ i : y_1 + \cdots + y_i \geq w_0 + w_1 + \cdots + w_l \right\}.$$

In other words, y_j is the first value where inequality (2) is violated if only $y_1, \ldots, y_{j-1}$ are used. (By definition, $y_j \in \{\varphi_0, \varphi_1, \ldots, \varphi_l\}$. In addition, if $y_j > \varphi_0$, then $y_{j+1} > y_j$.) From these properties we derive the following lemma, stating that among all sequences satisfying Eq. (2), y_i achieves the maximum values:

Lemma 5 *We have $d \leq c$ and $y_j \geq x_j$ for all $j \in \{1, \ldots, d\}$.*

Lemma 5 asserts that the greedy values $y_j \geq x_j$. This means AIDFS does not miss any potential compared to COPT.

- First phase: Traverse some light subtrees $T[e_{ij}]$ (set $H \subseteq \{1, \ldots, l\}$) in decreasing index order.
- Second phase: Traverse all vertices in $T[r']$ (routes containing r).
- Third phase: Traverse remaining light subtrees $T[e_{ij}]$ (set H) in increasing index order.

Let $z_1, \ldots, z_p$ be the lowest potentials reached by routes in the first phase of AIDFS(T_F), sorted in reverse order:

$$\varphi(r) \geq z_p \geq \cdots \geq z_1 > \varphi(r') = \varphi(0).$$

Lemma 6 *If $z_i \geq y_j$, then $z_{i+2} \geq y_{j+1}$.*

Lemma 6 establishes a crucial relationship: whenever an AIDFS route reaches a "sufficiently high" potential, then after 2 routes, we also reach the next potential in the greedy sequence y_j. Thus, combining Lemma 5 and Lemma 6 forms a key breakthrough in the proof - it allows us to control and bound the cost of the optimal COPT strategy by analyzing only the greedy AIDFS strategy, which is easier to construct and track.

Analysis of COPT(T) cost:

$$\xi(\text{COPT}(T_F)) = O_{\text{light}} + O_{\text{deep}} + O_{\text{path}} + O_{\text{flat}},$$

where:

- O_{light}: Cost on light subtrees $T[e_{ij}]$.
- O_{deep}: Cost on deep subtree $T[r']$.
- O_{path}: Cost on path P (excluding routes containing r').
- O_{flat}: Cost on path P for routes containing r'.

Cost of AIDFS(T_F):

$$\xi(\text{AIDFS}(T_F)) = D_{\text{light}} + D_{\text{deep}} + D_{\text{desc}} + D_{\text{flat}} + D_{\text{asc}},$$

where:

- D_{light}: Cost on light subtrees $T[e_{ij}]$.

- D_{deep}: Cost on heavy subtree $T[r']$.
- D_{desc}: Descending cost on P (first phase of AIDFS).
- D_{flat}: Cost on P for routes containing r' (second phase of AIDFS).
- D_{asc}: Ascending cost on P (final phase of AIDFS).

By Lemma 2 regarding cost on light subtrees D_{light}:

$$\xi(\text{AIDFS}(T[e_{ij}])) \leq 2\xi(\text{COPT}(T[e_{ij}])) - \vartheta(T[e_{ij}])).$$

Thus:

$$D_{\text{light}} \leq 2O_{\text{light}} - \vartheta(T_F)).$$

Let s be the minimal index for which $y_{s+1} > \varphi_0 = y_s$ and let $j_1, \ldots, j_q$, where $j_1 < j_2 < \cdots < j_q$, be the potentials reached by subsequent routes in AIDFS(T_F). Since $z_{j_1} \geq y_s$, by repeatedly applying Lemma 6, we obtain: $z_{j_i} \geq y_{s+\frac{i}{2}-1}$ for each $i \in \{1, \ldots, q\}$.

Descending and Ascending Costs ($D_{desc} + D_{asc}$) Descending cost (based on Lemma 5):

$$\begin{aligned} D_{\text{desc}} &= 2\sum_{i=1}^{q}(\varphi(T_F) - z_{j_i}) = 2(\varphi(T_F) - z_{j_1}) + 2\sum_{i=2}^{q}(\varphi(T_F) - z_{j_i}) \\ &= 2(\varphi(T_F) - \varphi_0) + 2\sum_{i=2}^{q}(\varphi(T_F) - z_{j_i}). \end{aligned}$$

Hence, we obtain:

$$\begin{aligned} D_{\text{desc}} &\leq 2\omega(P) + 2\sum_{i=2}^{q}(\varphi(T_F) - y_{s+[i/2]-1}) \\ &\leq 2\omega(P) + 4\sum_{i=2}^{q}(\varphi(T_F) - y_{s+i-1}) \\ &\leq 2\omega(P) + 4\sum_{i=2}^{q}(\varphi(T_F) - x_{s+i-1}) \leq 2\omega(P) + 2O_{\text{path}}. \end{aligned}$$

Similarly, the same bound holds for the ascending portion of AIDFS(T_F):

$$D_{\text{asc}} \leq 2\omega(P) + 2O_{\text{path}}.$$

Cost on heavy Subtree ($D_{deep} + D_{flat}$) By the induction hypothesis Theorem 2(ii):

$$D_{\text{deep}} < 12O_{\text{deep}} - 20\varphi(r).$$

Relationship with O_{flat}:

- Each COPT route through $T[r']$ traverse at least $2\omega(P)$, so: $O_{\text{flat}} \geq \frac{\varphi(r)}{\omega(P)} \cdot O_{\text{deep}}$.
- AIDFS has at most $\frac{D_{\text{deep}}}{2\varphi(r)} + 1$ routes containing r, so:

$$D_{\text{flat}} \leq 2\omega(P) \cdot \frac{D_{\text{deep}}}{2\varphi(r)} + 1 \leq 4\omega(P) + \frac{\omega(P) \cdot D_{\text{deep}}}{\varphi(r)}.$$

Combining these:

$$\begin{aligned}
D_{\text{deep}} + D_{\text{flat}} &\leq 4\omega(P) + \left(\frac{\omega(P)}{\varphi(r)} + 1\right) D_{\text{deep}} \\
&\leq 4\omega(P) + \left(\frac{\omega(P)}{\varphi(r)} + 1\right) (12 \cdot O_{\text{deep}} - 20\varphi(r)) \\
&\leq 4\omega(P) - 20(\omega(P) + \varphi(r)) + 12 \cdot O_{\text{deep}} \left(\frac{\omega(P)}{\varphi(r)} + 1\right) \\
&\leq 12O_{\text{deep}} + 12O_{\text{flat}} - 16\omega(P) - 20\varphi(r).
\end{aligned}$$

Combining all components:

$$\xi(\text{AIDFS}(T_F)) \leq 2O_{\text{light}} - \vartheta(T_F) + 4\omega(P) + 4O_{\text{path}} + 12O_{\text{deep}} + 12O_{\text{flat}} - 16\omega(P) - 20\varphi(r).$$

Simplifying using $\omega(P) + \varphi(r') = \varphi(r)$, we get:

$$\begin{aligned}
\xi(\text{AIDFS}(T_F)) &\leq 12O_{\text{deep}} + 12O_{\text{flat}} + 2O_{\text{light}} + 4O_{\text{path}} - 12\omega(P) - 20\varphi(r) - \vartheta(T_F) \\
&\leq 12\xi(\text{COPT}(T_F)) - 10\varphi(r) - \vartheta(T_F).
\end{aligned}$$

For heavydeg$(r) = 1$, we have thus proven that the AIDFS strategy's cost does not exceed 12 times the cost of the optimal COPT(T_F) strategy. This proof uses induction and detailed analysis of cost components in both AIDFS and COPT strategies. Specifically, while AIDFS may not be fully optimal, it remains near-optimal with a small error margin determined by factors like $\varphi(T_F)$ and $\vartheta(T_F)$.

The result shows that Theorem 2 holds for the initial Skinny Tree, and through Lemma 4 - a crucial technical tool for extending Theorem 2 to all trees.

The idea is to transform tree T_F without significantly altering exploration costs.

Definition of Potential Strategies and Selection of ε Denoted potential paths $R' = (L, v)$, where $L = (l_1, \ldots, l_p)$ is a leaf sequence of T_F and v is a vertex (either in T_F or T_ε). Each R' corresponds to an actual path in T_F by connecting the path from root to l_1, connecting consecutive leaves, then from l_p to v's nearest ancestor and back to the root.

Correspondingly, any DFS-B strategy on T_F can be represented as a sequence of potential paths. This concept enables "potential strategy mapping" - a uniform mapping of strategies between T_F and T_ε without examining individual paths.

The length of each potential path is computed, and conditions for selecting ε are established such that the sets of feasible strategies on T_F and T_ε coincide: for sufficiently small ε, a strategy feasible on T_F is also feasible on T_ε and vice versa. The deficiency $x(S)$ is defined for non-feasible strategies in T_F, and y is chosen based on $x(S)$ to ensure T_ε and T_F share the same space of feasible strategies.

Construction of T_ε With ε satisfying the above, we construct a new tree T_ε by "breaking" heavy edges where a parent node has only one heavy edge (similar to the Skinny Tree). Specifically, if at node v there is a heavy edge $\{u, v\}$ and v has $d > 1$ light child edges $e_1, \ldots, e_d$, we replace the heavy edge $\{u, v\}$ with a path of d consecutive small edges while maintaining the total length (initial edges of length $\varepsilon/(d-1)$, final edge of original length minus ε).

Simultaneously, each light edge e_i has its weight reduced by ε and is shifted down to level v_{i-1} (without increasing inter-node distances). This subdivision is applied to all similar cases in the tree. The result is that each node in T_ε has at most two child edges (at most one heavy and one light edge), satisfying the condition for Skinny Trees.

Thus T_ε satisfies the Skinny property: each segment connecting heavy edges has at most one light edge attached. This transformation technique is crucial, enabling the application of Theorem 2, originally defined only for Skinny Trees.

Analysis of T_ε :

Finally, we argue that for sufficiently small ε, transitioning to T_ε does not significantly alter the problem: the optimal costs $\mathrm{COPT}(T_F)$ and $\mathrm{COPT}(T_\varepsilon)$ are nearly equal, as are the worst-case costs $\mathrm{AIDFS}(T_F)$ and $\mathrm{ADFS}(T_\varepsilon)$. Through continuity arguments, we can ensure the cost difference remains small and the space of feasible strategies is preserved.

Combining these results shows: if Theorem 2 holds for T_ε (Skinny Tree), then it also holds for the original T_F. This completes the proof of Lemma 4.

Proof Conclusion Lemma 4 guarantees that for any tree T_F, we can transform it into T_ε satisfying the Skinny Tree (ST) property, and if Theorem 2 holds for T_ε, it also holds for T_F.

In the proof of Lemma 4, we have:

Lemma 7 *Some useful inequalities are derived as follows:*

- *(i)* $\xi(AIDFS(T_F)) \leq \xi(AIDFS(T_\varepsilon)) + 4\varepsilon n^2$,
- *(ii)* $\varphi(T_F) \leq \varphi(T_\varepsilon) + 2n\varepsilon$, *and*
- *(iii)* $\xi(COPT(T_\varepsilon)) \leq \xi(COPT(T_F))$.

Lemma 8 *ε always exists: $\mathcal{U} \neq \emptyset$.*

We may assume (by choosing ε sufficiently small, i.e., $\varepsilon < v(T_F)/2$) that all considered trees T_ε satisfy $v(T_\varepsilon) \geq v(T_F)/2$. Thus, $v(T_\varepsilon)$ is independent of how small $\varepsilon \in \mathcal{U}$ is chosen. Then, by Theorem 2(I):

$$\xi(\mathrm{AIDFS}(T_F)) \leq 12 \cdot \xi(\mathrm{COPT}(T_F)) - 10 \cdot \varphi(T_F).$$

By Lemma 8, there exists $\varepsilon \in \mathcal{U}$ such that $\varepsilon > 0$ and:

$$\varepsilon < \frac{v(T_F)}{12n^2},$$

ensuring T_ε satisfies the conditions of Lemma 8.

Then, by Lemma 7 (specifically, (i) yields the first inequality below, while (ii) and (iii) yield the fourth inequality below), we have:

$$\begin{aligned}\xi(\text{AIDFS}(T_F)) &\leq \xi(\text{ADFS}(T_\varepsilon)) + 4\varepsilon n^2 \leq 12 \cdot \xi(\text{COPT}(T_\varepsilon)) - 10 \cdot \varphi(T_\varepsilon) - \theta(T_\varepsilon) + 4\varepsilon n^2 \\ &\leq 12 \cdot \xi(\text{COPT}(T_\varepsilon)) - 10 \cdot \varphi(T_\varepsilon) - \frac{\theta(T_F)}{2} + 4\varepsilon n^2 \\ &\leq 12 \cdot \xi(\text{COPT}(T_F)) - 10 \cdot \varphi(T_F) + 2n\varepsilon - \frac{\theta(T_F)}{2} + 4\varepsilon n^2 \\ &\leq 12 \cdot \xi(\text{COPT}(T_F)) - 10 \cdot \varphi(T_F).\end{aligned}$$

Since $\theta(T_F) = 0$ by definition, this completes the proof.

A similar argument applies to Theorem 2 (ii) with the same ε. Therefore, Theorem 2 holds for T_F.

Similarly for case (ii), we have:

$$\xi(\text{AIDFS}(T_F)) \leq 12 \cdot \xi(\text{COPT}(T_F)) - 20 \cdot \varphi(T_F).$$

Since $\varphi(T_F) \geq 0$ (because $B/2 \geq$ height of T_F), we conclude:

$$\xi(\text{AIDFS}(T_F)) \leq 12 \cdot \xi(\text{COPT}(T_F)).$$

Hence:

$$\xi(\text{IDFS}(T_F)) \leq \xi(\text{AIDFS}(T_F)) \leq 12 \cdot \xi(\text{COPT}(T_F)).$$

This completes the proof that Theorem 2 holds for all trees T_F satisfying $B/2 \geq$ height of T_F.

Theorem 3 (Bound on the Number of Routes) *Given a forest T_F and an energy bound B such that the farthest leaf from the root is at most $B/2$, then the number of routes in IDFS satisfies:*

$$|IDFS(T_F)| \leq 12\,|\mathcal{R}|\,.$$

where $\mathcal{R}$ is the optimal strategy with the minimum number of routes.

Prove for Theorem 3

We have this observation:

$$|\mathcal{R}| \geq \left\lceil \frac{\xi(COPT(T_F))}{B} \right\rceil. \tag{3}$$

Proof the observation:

Each route $R_i \in \mathcal{R}$ has length $\ell(R_i) \leq B$, and hence the total cost satisfies

$$\xi(\mathcal{R}) = \sum_{i=1}^{|\mathcal{R}|} \ell(R_i) \leq |\mathcal{R}| \cdot B.$$

Rearranging gives the inequality: $|\mathcal{R}| \geq \frac{\xi(\mathcal{R})}{B}$.
Since $\mathcal{R}$ is optimal, we have $\xi(\mathcal{R}) = \xi(COPT(T_F))$, and thus: $|\mathcal{R}| \geq \frac{\xi(COPT(T_F))}{B}$.

Taking the ceiling on both sides yields

$$|\mathcal{R}| \geq \left\lceil \frac{\xi(COPT(T_F))}{B} \right\rceil.$$

To show the Invader in this algorithm, now we denote surplus $C = B - l(R_i)$. For each route R_i in $IDFS(T_F)$, after visiting the last vertex, return to the root, and ensure that each route in the $IDFS$ traversal (except the last route) reaches its maximum allowable length B. We have to use the surplus to explore the edges of other trees. If any edges in other trees satify: $2 \cdot \omega(\{v_{i(j+1)}, v_{(i+1)(j+1)}\}) > C$ where $v_{i(j+1)}$ is the node i^{th} of the $(j+1)^{th}$ trees(the other trees of the forest).

To facilitate analysis and proof of the Theorem 3, we need to transform the tree T_{j+1} into T'_{j+1} or T_F into T'_F(because when transforming the tree T_{j+1} into T'_{j+1}, mean that you are transform the large tree T_F) by splitting an edge.

In that case, we subdivide the edge $\{v_{i(j+1)}, v_{(i+1)(j+1)}\}$ into two edges by introducing a new vertex $x_{i(j+1)}$: $\{v_{i(j+1)}, x_{i(j+1)}\}$ with weight $\omega(\{v_{i(j+1)}, x_{i(j+1)}\}) = \frac{C}{2}$, and $\{x_{i(j+1)}, v_{(i+1)(j+1)}\}$ with weight

$$\omega(\{x_{i(j+1)}, v_{(i+1)(j+1)}\}) = \omega(\{v_{i(j+1)}, v_{(i+1)(j+1)}\}) - \frac{C}{2}.$$

This modification ensures the following properties:

Edge weight:

$$\omega(\{v_{i(j+1)}, x_{i(j+1)}\}) + \omega(\{x_{i(j+1)}, v_{(i+1)(j+1)}\}) = \omega(\{v_{i(j+1)}, v_{(i+1)(j+1)}\}).$$

Thus, the distances between original vertices are unchanged.

Route length: Each route in $IDFS(T'_F)$ (except possibly the last) ends exactly at the new vertex x_{ij} and has total length B.

Cost: The optimal traversal cost remains the same:

$$\xi(COPT(T'_F)) = \xi(COPT(T_F)).$$

Since the cost when exploring the T'_F is at most B for subsequence routes, except for the last route.

Route count:

$$|IDFS(T'_F)| = IDFS(T_F)|.$$

because the vertex visitation sequence remains unchanged except for the inclusion of $x_{i(j+1)}$.

Now we use the Theorem 1 for applying the cost approximation bound, the cost of the IDFS strategy satisfies:

$$\xi(IDFS(T_F)) \leq 12 \cdot \xi(COPT(T_F)).$$

Since the transformation from T_F to T'_F does not change the optimal cost, the same inequality holds for T'_F:

$$\xi(IDFS(T'_F)) \leq 12 \cdot \xi(COPT(T'_F)) = 12 \cdot \xi(COPT(T_F)). \tag{4}$$

In T'_F, all routes in $IDFS(T'_F)$ (except possibly the last) have length exactly B. Therefore, the number of routes is given by

$$|IDFS(T'_F)| = \left\lceil \frac{\xi(IDFS(T'_F))}{B} \right\rceil.$$

Using the inequality (4), we have:

$$|IDFS(T'_F)| \leq \left\lceil \frac{12 \cdot \xi(COPT(T_F))}{B} \right\rceil \leq 12 \cdot \left\lceil \frac{\xi(COPT(T_F))}{B} \right\rceil.$$

From observation (3), we have:

$$\left\lceil \frac{\xi(COPT(T_F))}{B} \right\rceil \leq |\mathcal{R}|,$$

where $\mathcal{R}$ is the optimal strategy. Thus:

$$|IDFS(T'_F)| \leq 12 \cdot |\mathcal{R}|.$$

Since $|IDFS(T'_F)| = |IDFS(T_F)|$, we conclude:

$$|IDFS(T_F)| \leq 12 \cdot |\mathcal{R}|.$$

2.2 Experimental problems

2.2.1 Exploration Enviroment

Each tree $T_i = (V_i, E_i, w_i)$ is a connected, weighted graph without cycles. An energy-constrained robot traverses all edges of each tree T_i and returns to the start location to recharge whenever its remaining energy is exhausted.

To facilitate the exploration, we attach roots r_i of T_i to *parent root*. For simplicity, we select the *parent root* as the root r_1 of the first tree (T_1), then by attaching each of the other r_i to this parent root. Therefore $T_F = (V, E, w)$:

$$V = \bigcup_{i=1}^{n} V_i, E = \left(\bigcup_{i=1}^{n} E_i \right) \cup \{(r_1, r_i) : i = 2, \dots, n\}, w = \left(\bigcup_{i=1}^{n} w_i \right) \cup \{(r_1, r_i) = 0 : i = 2, \dots, n\}.$$

The following example shows the consolidation of three independent trees into the large tree T_F. A similar method can be applied to the merging of n trees.

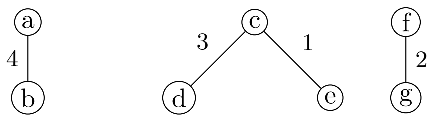


Fig. 3 A forest of three disjoint trees T_1, T_2, T_3.

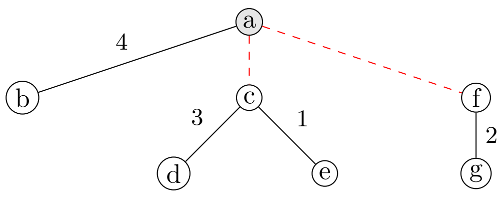


Fig. 4 Consolidation of three independent trees.

2.2.2 Traversal Process

Step 1:. Set v_{01}, the root of the first tree, as the parent root denoted r_1.

Step 2:. The route R_i continues from where the previous route R_{i-1} stopped. The route must satisfy the constraint:

$$d(r_1, v_{(i-1)j}) + l(v_{(i-1)j}, \dots, v_p) + d(v_p, r_1) \le B. \tag{5}$$

The farthest node v_p that satisfies Eq. (5) is denoted as v_{ij} where edge i^{th} corresponds to j^{th}-tree.

Step 3:. After R_i satisfies the constraint in the Equation 5, compute the surplus as: $C = B - \xi(R_i)$. Use this surplus to explore nodes from other trees, selecting those that satisfy:

$$2 \cdot \sum_{i,j}^{n} w_{ij} \le C,$$

where w_{ij} is the weight of the edge with the i^{th} edge of j^{th} tree. The additional route explored using this surplus is denoted R_C.

Step 4:. Let $R_i = P_{i-1} \circ (v_{(i-1)j}, \ldots, v_{ij}) \circ P_i^R \circ R_C$, where P_{i-1} is the path from r_1 to $v_{(i-1)j}$; P_i^R is the path from v_{ij} back to r_1.

2.2.3 Complexity of the Algorithm

We analyze the complexity in terms of the following components:

The cost of performing IDFS on each tree:. Assume a forest consisting of n disjoint trees $T_1, T_2, \ldots, T_n$, where each tree T_i contains $n_i = |V_i|$ nodes and has a diameter D_i. Let B be the maximum energy budget allowed per traversal. Due to the energy constraint B, the traversal is partitioned into multiple segments. The total energy required to complete a IDFS on the tree is $O(n \cdot D)$ in the worst case (each node visited at depth up to D). Since each segment is limited to an energy budget B, performing $IDFS$ on each tree is given by:

$$O\left(n_i \cdot \frac{D_i}{B}\right).$$

Therefore, the total complexity of traversing all n trees independently is:

$$\sum_{i=1}^{n} O\left(n_i \cdot \frac{D_i}{B}\right).$$

If we denote the total number of nodes as $\sum_{i=1}^{n} n_i$, then the worst-case overall complexity for separately traversing all trees becomes:

$$O\left(\sum_{i=1}^{n} n_i \cdot \frac{D_i}{B}\right).$$

Tree Merging. To potentially improve efficiency, the forest can be connected into a single tree by adding $n-1$ edges. Let D be the diameter of the resulting merged tree T_F, and $N = \sum_{i=1}^{n} n_i$ the total number of nodes.

Additionally, let the merging process involve h connection operations (up to $n-1$), each incurring a merging cost of $f(N)$. The total merging cost is:

$$O(h \cdot f(N)) = O((n-1) \cdot f(N)).$$

Hence, the total complexity including merging and traversal is:

$$O\left(N \cdot \frac{D}{B} + (n-1) \cdot f(N)\right).$$

In practice, $f(N)$ is typically $O(1)$ or $O(N)$ depending on the merging mechanism. If merging cost is $f(N) \leq \frac{D}{B}$, the dominating term remains the traversal:

$$O\left(N \cdot \frac{D}{B}\right).$$

Surplus Exploration. Since no new node in the current subtree can be reached, at the route R_i, after returning to the root, the surplus is $C_i = B - \ell(R_i)$ Thus, we would still be enough budget to traverse at least one edge or set of edges unexplored of weight at most $B/4$ The algorithm then uses this surplus C_i to perform an IDFS. By IDFS analysis, the cost of exploring by surplus:

$$O\big(n'_i \times \tfrac{D}{C_i}\big),$$

where n'_i is the number of nodes that surplus C_i can explore;

If we denote the total number of nodes explored by C_i as $N' = \sum_{i=1}^{n-1} n_i$, then the overall complexity for separately traversing trees by surplus becomes:

$$O\left(\sum_{i=1}^{n-1} N' \cdot \frac{D}{C_i}\right).$$

Since $C_i < B/2$, yielding

$$O\Big(N' \times \frac{D}{C_i}\Big) = O\Big(N' \times \frac{(D)}{(B/2)}\Big) = O(N' \times \frac{2D}{B}).$$

Thus, across all segments, surplus-driven exploration contributes only linear time. Combining the above factors, the total time complexity of the algorithm is:

$$O\left(\frac{2 \cdot N' \cdot D}{B} + \frac{N \cdot D}{B}\right) = O\left((2 \cdot N' + N)\frac{D}{B}\right).$$

3 Deployed Algorithm

The following examples illustrate the IDFS algorithm for exploring a forest comprising n trees, which have been merged according to the methodology described in Section 2.2.2. Exploring a forest of n trees with energy constraints by giving a real number $B \geq 1$, and a forest composed of n trees, where the maximum distance from the root to any leaf in each tree is at most $B/2$, we conduct the exploration of this forest using the $IDFS$ algorithm. The following examples are provided for illustration.

Example 1 Considering a forest includes four trees shown in Figure 5. We summarize the information and a comparison shown in Table 1 and Table 2.

Example 2 Considering a forest comprises five trees depicted in Figure 6. We summarize the information, and a comparison is shown in Table 3 and Table 4, respectively. Table 5 and Table 6 present a comparative analysis of total costs and the number of routes, surplus, and utilization costs.

4 Discussion And Limitation

In this study, we introduce the IDFS algorithm for traversing a forest composed of n independent trees. Our key contributions include several performance improvements over traditional approaches. First, we achieve route reduction, lowering the total number of routes by up to 20% compared to applying the classical PDFS algorithm to each tree individually. Second, we demonstrate cost savings, with traversal costs decreasing by an average of up to 30% thanks to the reuse of surplus energy from previous routes. Third, the algorithm provides improved continuity, allowing exploration across multiple trees within a single traversal session, rather than requiring resets between trees. Finally, we observe that surplus energy is more effectively utilized, significantly reducing unnecessary traversal and enabling further cost savings.

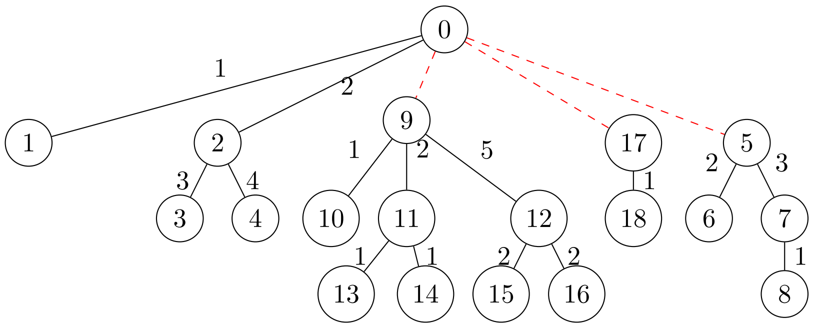


Fig. 5 Illustration of the four-tree forest with traversal routes.

Despite these advantages, the algorithm still has limitations. It relies on prior knowledge of a fixed energy budget B, and assumes that all leaf nodes are within a distance of $B/2$ from the root, making it less applicable in dynamic or unknown-B environments. Additionally, IDFS assumes an acyclic input structure—a forest of trees—and cannot be directly applied to general graphs containing cycles, which would require fundamentally different strategies. Lastly, in the worst-case scenarios, the algorithm may not fully optimize redundancy. For instance, if a tree's weight exceeds the redundancy allowance per route, or if redundancy is triggered at inefficient times, the result is repeated edge visits and increased traversal cost. Similarly, if the parameter B is exhausted too early or is too large relative to the tree size, no effective route reduction may occur. To address these limitations, future work will focus on three directions. We aim to explore online adaptations of the algorithm that can handle unknown or variable B. We also plan to extend IDFS to graphs with cycles,

Table 1 Simulation of *IDFS* for four trees that union to the first tree.

Route	Node-Sequence	Cost
1	$0 \to 1 \to 0 \to 2 \to 3 \to 2 \to 0 \to 5 \to 6 \to 5 \to 0 \to 9 \to 10 \to 9 \to 0$	18
2	$0 \to 2 \to 4 \to 2 \to 0 \to 5 \to 7 \to 5 \to 0$	18
3	$0 \to 5 \to 7 \to 8 \to 7 \to 5 \to 0 \to 9 \to 11 \to 13 \to 11 \to 14 \to 11 \to 9 \to 0 \to 17 \to 18 \to 17 \to 0$	18
4	$0 \to 9 \to 12 \to 15 \to 12 \to 16 \to 12 \to 9 \to 0$	18

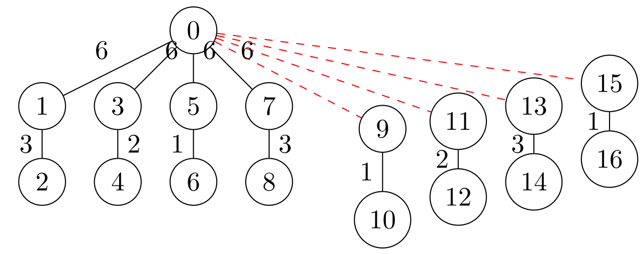


Fig. 6 Illustration of the five-tree forest with traversal routes.

Table 2 Simulation for four-trees with the first traverse.

Route	Node-Sequence	Cost
1	$0 \to 1 \to 0 \to 2 \to 3 \to 2 \to 0$	12
2	$0 \to 2 \to 4 \to 2 \to 0 \to 9 \to 10 \to 9 \to 11 \to 9 \to 0$	18
3	$0 \to 9 \to 11 \to 13 \to 11 \to 14 \to 11 \to 9 \to 12 \to 9 \to 0$	18
4	$0 \to 9 \to 12 \to 16 \to 12 \to 15 \to 12 \to 9 \to 0 \to 17 \to 0$	18
5	$0 \to 17 \to 18 \to 17 \to 0 \to 5 \to 6 \to 8 \to 6 \to 5 \to 7 \to 5 \to 0$	14

Table 3 Simulation of *IDFS* for five trees that union to the first tree.

Route	Node-Sequence	Cost
1	$0 \to 1 \to 2 \to 1 \to 0 \to 9 \to 10 \to 9 \to 0$	20
2	$0 \to 3 \to 4 \to 3 \to 0 \to 11 \to 12 \to 11 \to 0$	20
3	$0 \to 5 \to 6 \to 5 \to 0 \to 13 \to 14 \to 13 \to 0$	20
4	$0 \to 7 \to 8 \to 7 \to 0 \to 15 \to 16 \to 15 \to 0$	20

generalizing the current tree-based strategy. Finally, we will investigate algorithmic enhancements to improve performance in edge cases and further minimize redundancy

Table 4 Simulation for five trees with the first traverse.

Route	Node-Sequence	Cost
1	$0 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 0$	18
2	$0 \rightarrow 3 \rightarrow 4 \rightarrow 3 \rightarrow 0$	16
3	$0 \rightarrow 5 \rightarrow 6 \rightarrow 5 \rightarrow 0$	14
4	$0 \rightarrow 7 \rightarrow 8 \rightarrow 7 \rightarrow 0 \rightarrow 9 \rightarrow 10 \rightarrow 9 \rightarrow 0$	20
5	$0 \rightarrow 11 \rightarrow 12 \rightarrow 11 \rightarrow 0 \rightarrow 13 \rightarrow 14 \rightarrow 13 \rightarrow 0 \rightarrow 15 \rightarrow 16 \rightarrow 15 \rightarrow 0$	12

Table 5 Comparison of four trees.

Criteria	**Our Result**	**Traditional PDFS**
Number of Routes	4	5
Total Cost	72	90
Utilized Cost	72	80
Surplus	0	10

Table 6 Comparison of five trees.

Criteria	**Our Result**	**Traditional PDFS**
Number of Routes	4	5
Total Cost	80	100
Utilized Cost	80	80
Surplus	0	20

5 Conclusion And Future Work

Through analysis of the exploration environment, theoretical proofs, and the traversal process, we have demonstrated the precision of the proposed algorithm. Specifically, the algorithm demonstrates its optimality by exploring a forest of n trees, thereby reducing both the number of required routes and the total energy consumption. The proposed algorithm correctly fulfills the original objective of solving the traversal problem in a forest of n trees. In particular, the paper sequentially demonstrates its application to four, and five trees. The results show that both the number of traversal paths and the associated cost are optimized compared to traversing a single tree. However, certain limitations remain: In some cases, the number of routes and the total energy cost do not improve significantly compared to applying the traditional PDFS algorithm independently on each tree. In future work, we aim to develop an improved version of the algorithm that addresses these limitations more effectively, combining machine learning [9]. In addition, we plan to deploy the algorithm in practical applications, such as exploration of oil pipeline systems or treasure hunting in harsh environments where maps are already available.

Delarations

Authors' contributions: Bui Phi Hung, Nguyen Minh Tuan: Conceptualization, data curation, investigation, methodology, software, visualization, writing-original draft and writing-review and editing, validation, visualization, writing-original draft and writing-review and editing.

Availability of data and materials: The author declares no conflict of interest.

Competing interests: The authors confirm that there is no conflict of interest.

Funding: Not applicable

Acknowledgments: The authors acknowledge anonymous referees' comments for worthy advice and valuable discussions.

References

[1] Das, S., Dereniowski, D., Uznański, P.: Energy constrained depth first search. Algorithmica **86**(12), 3759–3782 (2024) https://doi.org/10.1007/s00453-024-01275-8

[2] Tuan, N.M., Son, N.H., Van, C.V.: Piecemeal-DFS algorithm for energy-constrained depth-first search on the tree-merging problem. In: Proceedings of the 2nd National Symposium on Natural Sciences and Applications in the Digital Age (NSA) (In Vietnamese), ISBN: 978-604-67-3330-0 2025, (2025)

[3] Korf, R.E.: Depth-first iterative-deepening. Artificial Intelligence **27**(1), 97–109 (1985) https://doi.org/10.1016/0004-3702(85)90084-0

[4] GeeksforGeeks: Depth Limited Search. Accessed June 2025. https://www.geeksforgeeks.org/depth-limited-search/

[5] Bampas, E., Chalopin, J., Das, S., Hackfeld, J., Karousatou, C.: Maximal Exploration of Trees with Energy-Constrained Agents. arXiv preprint arXiv:1802.06636 (2018). https://arxiv.org/abs/1802.06636

[6] Dutta, A., Sharma, G.: A Constant-Factor Approximation Algorithm for Online Coverage Path Planning with Energy Constraint. arXiv preprint arXiv:1906.11750 (2019). https://arxiv.org/abs/1906.11750

[7] Banerjee, N., Chakraborty, S., Raman, V.: Improved space efficient algorithms for bfs, dfs and applications. In: Lecture Notes in Computer Science, pp. 119–130 (2016). https://doi.org/10.1007/978-3-319-42634-1_10

[8] Tarjan, R.E., Zwick, U.: Finding strong components using Depth-First search. arXiv preprint arXiv:2201.07197 (2022). https://arxiv.org/abs/2201.07197

[9] Tuan, N.M., Meesad, P., Nguyen, H.H.C.: English–vietnamese machine translation using deep learning for chatbot applications. SN Computer Science **5**(1), 5 (2024) https://doi.org/10.1007/s42979-023-02339-2